ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ
INFORMATION TECHNOLOGY, COMPUTER SCIENCE, AND MANAGEMENT



УДК 004.652 DOI 10.12737/19696

Технология раздельного формирования многомерных данных*

С. В. Зыкин[1], С. В. Мосин[2], А. Н. Полуянов[3]**
[1, 2, 3]Институт математики им. С. Л. Соболева СО РАН, г. Новосибирск, Российская Федерация

Technology of separate generation of multidimensional data ***

S. V. Zykin[1], S. V. Mosin[2], A. N. Poluyanov[3]**
[1, 2, 3]Sobolev Institute of Mathematics, Novosibirsk, Russian Federation

Предметом исследования является технология формирования многомерного представления данных с использованием раздельного задания размерностей и мер. Цель — обеспечение максимального уровня автоматизации работы пользователей при формировании новых кубов данных. В ходе проведенных изысканий решены следующие задачи: определена последовательность формирования промежуточных представлений данных; исследована корректность этих представлений; разработаны эффективные алгоритмы формирования представления и проверки корректности. Теоретической основой исследования являются методы межмодельных преобразований данных. При этом в качестве исходной модели данных используется классическая реляционная модель, в качестве целевой — расширенная модель многомерных данных с несбалансированными иерархиями в размерностях. В результате проведенной работы представлена технология формирования многомерных данных. Полученные результаты могут использоваться аналитическими службами различных предприятий в процессе обработки значительных объемов данных. Предложенная технология формирования многомерных данных является развитием традиционных OLAP-технологий.

Ключевые слова: гиперкуб, реляционная база данных, OLAP.

The research subject is the technology of generating the multivariable data representation with using separate formation of dimensions and measures. The purpose of the study is to provide a full level of the user's work automation at the formation of new data cubes. In the course of investigation, the following problems are solved: the sequence of generating intermediate data representations is determined; the correctness of these representations is studied; efficient algorithms for generating the representations and checking their correctness are developed. The theoretical basis is the methods of the inter model mapping. Herewith, a classical relational model is used as a source data model, an extended model of the multidimensional data with unbalanced hierarchies in dimensions – as a target one. The work result is the multidimensional data construction technology. Consequently, the results obtained can be used by the analytical departments at various enterprises in processing large data volumes. The proposed technology of the multidimensional data formation is the traditional OLAP-technologies development.

Keywords: hypercube, relational database, OLAP.

Введение. Исследование OLAP (online analytical processing — аналитическая обработка в реальном времени) предполагает рассмотрение свойств моделей гиперкубов [1–3] и операций их преобразования [2, 4] с целью анализа данных. Особое внимание уделяется построению иерархий в размерностях [2, 3, 5–7], что позволяет гарантировать корректность операций агрегации данных. В работах [3, 5, 7] рассматриваются нормальные формы для многомерных моделей данных, которые позволяют контролировать неопределенные значения (NULL) в иерархиях размерностей.

В большинстве работ предполагается, что кубическое представление данных должно быть постоянно хранимым и периодически обновляемым из операционной базы данных (MOLAP — многомерная OLAP) для минимального времени отклика системы на запросы пользователя. Другой подход заключается в динамическом формировании многомерных данных с преобразованием схемы исходной операционной базы данных в «звезду» или «снежинку» (ROLAP — реляционная OLAP). Общий недостаток этих двух подходов — регламентированность предполагаемых операций анализа данных.

---

*Работа выполнена в рамках инициативной НИР.
** E-mail: szykin@mail.ru, svmosin@gmail.com, andrey.poluyanov@gmail.com
*** The research is done within the frame of the independent R&D.

В данной статье предполагается, что аналитическая работа пользователя основана на необходимости формирования новых гиперкубов из исходного реляционного представления данных.

Рассмотрим формализацию задачи. Пусть задана схема базы данных $\Re = \{R_1, R_2, ..., R_k\}$, полученная в результате нормализации отношений [8, 9]. Отношения R_i определены на множестве атрибутов $U = \{A_1, A_2, ..., A_n\}$. Пусть $[R_i]$ — схема отношения, множество атрибутов, на которых определено отношение R_i. Предположим, что схема $\Re$ является редуцированной [9], то есть не существует двух отношений таких, что $[R_i]\subseteq[R_j]$, при $i \neq j$. Кортеж $t\,[X]$ — совокупность значений атрибутов $A_j\in X\subseteq[R_i]$, заданных в кортеже $t\in R_i$. Неопределенное значение *NULL* атрибута A_j в кортеже t: $t\,[A_j] = NULL$ не равно любому другому значению, в том числе другому неопределенному значению.

Многомерное представление будем задавать в виде совокупности размерностей $\{D_1,D_2,\dots,D_d\}$, где D_l — множество расширенных имен атрибутов: $R_i\,A_j$, $A_\varphi\in[R_i]$; M — множество мер, также заданных в виде расширенных имен атрибутов. Значения D_l являются значениями координат гиперкуба, значения M будут располагаться в рабочей области гиперкуба. Для каждой размерности задается ограничение в виде логической формулы F_l.

В данной работе предлагается отказаться от необходимости выполнения функциональной зависимости [3, 5, 7, 10, 11]

$$D_1D_2\dots D_d \to M, \quad (1.1)$$

которая означает, что любому составному вектору значений размерностей $D_1D_2\dots D_d$ соответствует не более одного вектора значений мер M.

Отказ от зависимости (1.1) позволит использовать содержательные (не ключевые) атрибуты в размерностях и иметь в одной ячейке гиперкуба несколько значений (список) атрибута $R_i\,A_j\in M$. Списки значений используются в анализе данных, когда значения параметров не надо соотносить с объектами.

Пример 1. Рассмотрим фрагмент базы данных лечебного заведения. Задано множество атрибутов: A_1 — № пациента, A_2 — ФИО пациента, A_3 — № показателя, A_4 — показатель, A_5 — значение показателя, A_6 — № дня получения показателя, A_7 — группа пациентов. На предложенном множестве атрибутов существуют следующие зависимости: $DEP = \{A_1\to A_2A_7,\ A_3\to A_4,\ A_1A_3A_6\to A_5\}$. По правилам построения нормальных форм [8, 9] будет получена следующая схема базы данных:

— пациенты — $R_1\,(\underline{A_1}, A_2, A_7)$;
— перечень анализов — $R_2\,(\underline{A_3}, A_4)$;
— результаты анализов — $R_3\,(\underline{A_1}, \underline{A_3}, \underline{A_6}, A_5)$.

Одно из возможных представлений гиперкуба приведено в табл. 1.

Таблица 1

Сводная таблица анализов пациентов

Показатель	Креатинин		Белок		Билирубин	
Группа пациентов	2	3	2	3	2	3
№ дня получения показателя	Значение показателя					
1	61,97,78,101...	64,104,69,49...	82,70,67,69...	70,64,80,74...	14.2,17.8,18.84,44.3...	19.5,16.8,8.6,19.5...
2	63,102,83,113...	71,108,71,32...	64,58,68,61...	55,57,54,62...	34.7,15.4,96.5,64.9...	36.8,19.5,32.4,73.9...
3	59,59,87,79...	71,110,75,51...	68,62,58,59...	55,65,70,65...	19.5,17.8,83.78,114.3...	24.9,12.3,15.8,30.3...

В табл. 1 атрибуты размерностей представлены жирным шрифтом, атрибуты мер — курсивом, значения атрибутов — обычным шрифтом.

Схема гиперкуба в табл. 1 может быть представлена в следующем виде:

$$\{R_3\,A_6\} \times \{R_2\,A_4\{R_1\,A_7\,(R_3\,A_5)\}\}, \quad (1.2)$$

где $D_1 = \{R_3\,A_6\}$ и $D_2 = \{R_2\,A_4,\ R_1\,A_7\}$ — размерности; $M = \{R_3\,A_5\}$ — мера.

Логическое ограничение:

$F = (R_2\,A_4 =$ ' Креатинин ' $\vee\ R_2\,A_2 =$ ' Белок ' $\vee R_2\,A_2 =$ 'Билирубин'$)\wedge\ (R_1A_7 = 2 \vee R_1\,A_7 = 3)$.

При формировании представлений в примере использовались так называемые контексты и таблица соединения, определение и способ формирования которых рассматриваются далее.

2. Технология формирования многомерных данных. Для автоматизации построения представления многомерных данных используется следующая детализация последовательности их формирования, предложенная в работе [11].

1. Из списка атрибутов БД пользователь формирует множества атрибутов размерностей $D_1, D_2, ..., D_d$ и меры M. Естественными являются ограничения: $D_i \cap D_j = \varnothing, i \neq j, D_i \cap M = \varnothing, i, j = 1, 2, \ldots, d$.

2. Формирование иерархий размерностей для множеств атрибутов $D_1, D_2, ..., D_d$. Иерархии формируются автоматически по правилам, рассмотренным в работе [12], и при желании пользователь может их модифицировать.

3. По шаблону, соответствующему дизъюнктивной нормальной форме, задаются логические ограничения на размерности $F_1, F_2, \ldots, F_d$, где логическая формула F_i задана на атрибутах размерности D_i.

4. Формирование контекстов размерностей $C_1, C_2, \ldots, C_d$ (некоторые контексты могут быть пустыми, а некоторые — псевдоконтекстами). Далее будут представлены соответствующие определения.

5. Формирование контекста приложения C_0 и соответствующей реализации в виде таблицы соединений TJ_0.

6. Формирование реализаций размерностей $TJ_1, TJ_2, \ldots, TJ_d$ с сортировкой значений в соответствии с иерархией. Если контекст размерности не пуст, то он используется для формирования TJ_i, в противном случае реализация размерности является проекцией TJ_0.

7. Формирование реализации (представления) многомерной таблицы (заполнение значений мер на соответствующих местах таблицы). Если в одной ячейке рабочей области гиперкуба значений несколько, то они перечисляются через запятую.

Пользователь вручную выполняет шаги 1 и 3 и осуществляет выбор предложенных вариантов в шагах 2, 4 и 5. Все остальные операции выполняются автоматически.

3. Контексты

3.1. Свойства контекстов. Пусть DEP — множество зависимостей (функциональных, многозначных, включения, соединения), определенных на множестве атрибутов U и множестве отношений $\mathfrak{R}$. Пусть R — отношение, определенное на множестве атрибутов U (универсальное реляционное отношение). В работе используются классические определения зависимостей: функциональных, многозначных, зависимостей соединения, реализованных зависимостей [8, 9], зависимостей включения [13, 14, 15].

Пусть $C = \{R_1, R_2, ..., R_q\}$ — произвольное подмножество отношений реляционной БД.

Определение 3.1. Множество C будем называть контекстом, если оно удовлетворяет свойству соединения без потери информации (СБПИ) на зависимостях DEP, реализованных в C.

Замечание. В основе контекста лежит операция естественного соединения, которая собирает из различных отношений БД связанные друг с другом по значению данные.

Алгоритм проверки свойства СБПИ [8] является полиномиальным, но все равно довольно затратным по времени и по памяти для больших схем БД. Рассмотрим вспомогательные свойства, которые позволят улучшить эти характеристики.

Теорема 3.1. Множество отношений C обладает свойством СБПИ, если существует отношение $R_i \in C$, замыкание первичного ключа которого совпадает со всем множеством атрибутов отношений множества C.

Пусть $C_m = \{R_1, R_2, ..., R_q\}$ — произвольное множество отношений и $[C_m] = [R_1] \cup [R_2] \cup ... \cup [R_q]$.

Теорема 3.2. Множество отношений $C_{m+1} = \{R_1, R_2, ... , R_q, R_{q+1}\}$ не обладает свойством СБПИ на DEP, если зависимость $Z \twoheadrightarrow X(Y)$ не выводима из DEP, где $X \subseteq [C_m]$, $Y \subseteq [R_{q+1}]$ и $[C_m] \cap [R_{q+1}] \subseteq Z$.

Определение 3.2 (существующее соединение). Выражение $R_1 \bowtie R_2 \bowtie \ldots \bowtie R_q$ будем называть существующим соединением, если для совокупности отношений R_i, $I = 1,\ldots,q$, существует хотя бы одна перестановка $V_1, V_2, ..., V_q$ отношений $R_1, R_2, \ldots, R_q$ такая, что

$$([V_1] \cup [V_2] \cup \ldots \cup [V_j]) \cap [V_{j+1}] \neq \varnothing, j = 1, \ldots, q–1.$$

Теорема 3.3. Если множество отношений $C = \{R_1, R_2,\ldots, R_q\}$ не образует существующее соединение, то оно не обладает свойством СБПИ на множестве функциональных зависимостей FD.

Проверки условий, предложенных в теоремах 3.1–3.3, являются менее затратными по памяти и по времени, чем проверка свойства СБПИ по алгоритму [8]. Осталось разобраться со сложностью алгоритма проверки свойства существующего соединения.

Вход: множество отношений $C = \{R_{mas[1]}, R_{mas[2]},\ldots, R_{mas[p]}\}$, mas — массив с номерами отношений, для которых осуществляется проверка существования соединения. Считаем, что элементы массива нумеруются с 1, p — число элементов массива.

Выход: flag_exist = 1 — если соединение существует, flag_exist = 0 — если соединение не существует.

Теорема 3.4. На выходе алгоритма (рис. 1) flag_exist = 1 тогда и только тогда, когда множество отношений $C = \{R_{mas[1]}, R_{mas[2]},\ldots,R_{mas[p]}\}$ образует существующее соединение.

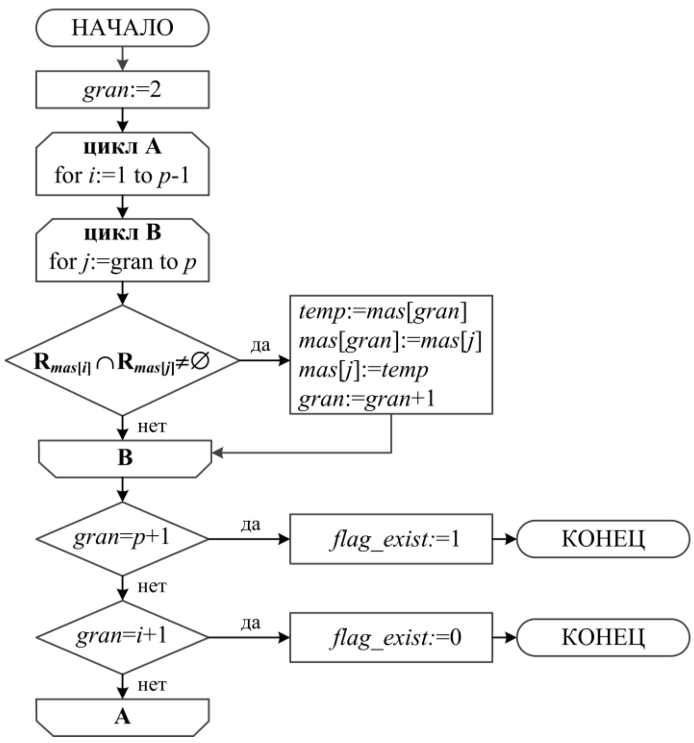


Рис. 1. Блок-схема алгоритма проверки существования соединения

С формальной точки зрения разработанный алгоритм является аналогом алгоритма поиска в ширину на графах [16]. Отличие в том, что на вход рассмотренного алгоритма подается не полностью построенный граф, а только его вершины, при работе алгоритма строятся только те ребра, которые необходимы для проверки свойства существования соединения отношений.

Анализ рассмотренного алгоритма показал, что самая трудоемкая операция — проверка пересечений. Она в алгоритме выполняется $p(p-1)/2$ раз. Аналогичные по сложности операции при проверке свойства СБПИ выполняются примерно $np(1+p)$ раз, где n — общее количество атрибутов в схеме базы данных. При этом учитывалось, что минимальное покрытие множества функциональных зависимостей [8] по мощности примерно равно количеству отношений. При проверке свойства СБПИ могут использоваться многозначные зависимости и зависимости соединений. Поскольку n не менее чем на порядок больше p и примерно 70 процентов комбинаций отношений на схеме БД не образуют существующее соединение, то применение рассмотренного алгоритма перед проверкой свойства СБПИ дает существенный выигрыш в результирующем количестве операций.

3.2. Формирование контекстов. Первоначальный выбор размерностей и мер гиперкуба предлагается сделать в расширенном виде: $R_i A_j$, где R_i — наименование отношения из исходной реляционной БД, A_j — наименование атрибута в этом отношении. Таким образом будет задано начальное множество отношений $C^0 = \{R^0{}_1, R^0{}_2, ..., R^0{}_q\}$, участвующее в обязательном порядке сначала в формировании таблицы соединения, а потом — гиперкуба.

Совокупность отношений, по которым строится гиперкуб, должна удовлетворять свойству СБПИ [17], поскольку лишние кортежи в промежуточном представлении данных дают лишние значения в рабочей области гиперкуба. Следовательно, дальнейшая задача состоит в дополнении множества C^0 отношениями из $\Re$, чтобы результирующее множество отношений удовлетворяло свойству СБПИ на множестве зависимостей, то есть являлось контекстом. В общем случае таких вариантов дополнения существует несколько. В работе [11] установлены критерии, позволяющие сделать перебор отношений направленным.

— Замыкание первичного ключа нового отношения R_i совпадает со всем множеством атрибутов в выбранных отношениях. Дополнение этого отношения к C^0 гарантирует выполнение свойства СБПИ по теореме 3.1. Такое отношение получает приоритет 3.

— Для отношения R_i выполнено условие существования связи, соответствующей $R_i[X]\subseteq R_j[X]$ с уже выбранными отношениями R_j, где множество атрибутов X является первичным ключом отношения R_j. Такое отношение получает приоритет 2, поскольку высока вероятность выполнения свойства СБПИ для результирующего множества отношений.

— Если дополняемое отношение R_i не удовлетворяет условиям теорем 3.2 и 3.3, то такое отношение получает приоритет 1. Остальные отношения получают приоритет 0.

— Формируемый контекст не должен содержать лишних отношений, наличие которых обусловлено только порядком присоединения отношений к контексту в алгоритме.

Перечисленные критерии увеличивают вероятность более быстрого достижения результата.

Введем обозначения. Пусть $C^1 = \{R^1{}_1, R^1{}_2, ..., R^1{}_p\}$ — множество отношений, не входящих в исходное множество $R^0 : R^1 = \Re \setminus R^0$. U^0 — множество атрибутов, на котором определены отношения из $R^0 : U^0 = [R^0{}_1]\cup[R^0{}_2]\cup\ldots\cup[R^0{}_q]$. DEP^0 — множество зависимостей, реализованных на отношениях из R^0.

В работе [11] рассмотрен алгоритм формирования контекстов, основанный на последовательном формировании сочетаний отношений и проверке их на свойство СБПИ.

Далее контекст приложения будем обозначать C_0.

В примере 1 встречаются ячейки с несколькими значениями (списком). Возникает закономерный вопрос: если в одном списке присутствуют два и более совпадающих значения, то дублируют ли они друг друга. Ответ следующий: если эти значения соответствуют одному и тому же объекту и одному и тому же параметру, то это дублирование. В результирующем представлении гиперкуба *GC* идентификаторы объектов отсутствуют, однако они есть в промежуточном представлении данных — таблице соединений *TJ*. Для того чтобы в дальнейшем иметь возможность определять дублированные значения, введем понятие ключа меры.

Определение 3.3. Множество атрибутов KM_j будем называть ключом атрибута меры $R_iA_j\in M$ в таблице соединений *TJ*. Если $KM_j \subseteq [TJ]$, зависимость $KM_j \rightarrow R_i A_j$ выводима на множестве функциональных зависимостей, и не существует выводимой зависимости $Y \rightarrow R_i A_j$, где $Y\subset KM_j$. Пусть $KM = KM_1\cup KM_2\cup\ldots\cup KM_h$, где $h = |M|$ — общий ключ для всех мер гиперкуба.

После того как сформирована схема *GC* с установлением иерархий в размерностях и присоединением мер к атрибутам одной из размерностей (в примере 1 меры присоединены к вертикальным размерностям), простейшим решением задачи формирования гиперкуба по контексту приложения $C_0=\{R'_1, R'_2, ..., R'_p\}$ при $F=\varnothing$ является выполнение операции естественного соединения отношений для формирования промежуточного представления *TJ*:

$$\mathrm{TJ} = \mathrm{R}'_1[\mathrm{V}_1] \bowtie \mathrm{R}'_2[\mathrm{V}_2] \bowtie\ldots\bowtie \mathrm{R}'_p[\mathrm{V}_p]. \quad (3.1)$$

Здесь *p* — количество отношений в контексте приложения; V_i — множество атрибутов $A_j\in[R'_i]$, для которых: либо существует размерность D_l — такая, что $R'_iA_j \in D_l$, либо $R'_iA_j\in M$, либо $R'_iA_j\in KM$, либо существует $R'_v\in C_0$ — такое, что $A_j\in[R'_v]$ и $i\neq v$, либо существует логическая формула F_l и $A_j\in[F_l]$.

Для преобразования в гиперкуб значения размерностей формируются в виде проекций по соответствующим атрибутам: $TJ[D_l]$ с необходимой сортировкой кортежей каждой размерности в соответствии с иерархией.

Завершается построение *GC* присваиванием значений мер *M* в рабочей области *GC*: для каждого кортежа $t\in TJ$ на пересечении значений координат $t[D_l]$, $l=1,\ldots,d$ ставится значение $t[A_j]$, $R'_iA_j\in M$. Во всех остальных ячейках *GC* ставится значение *NULL*. Проблема дублированных значений в ячейках гиперкуба и остальные детали реализации технологии будут рассмотрены далее.

Предложенная процедура формирования *GC* [11] не решает следующие проблемы.

— Если какому-либо набору значений размерности не соответствует ни одно значение меры, то эти значения размерностей не появятся в реализации гиперкуба (по свойству операции естественного соединения). Однако отсутствие значений мер также является предметом анализа данных.

— Ограничения на значения размерностей могут быть заданы опосредованно — через значения на связанные кортежи в отношениях, которые не входят в контекст приложения. Тогда эти отношения должны образовывать отдельный контекст с отношениями для размерностей.

— Для некоторых размерностей необходимо иметь Декартово произведение исходных отношений (все возможные комбинации атрибутов одной размерности). Тогда эти отношения не должны дополняться другими отношениями для получения контекста.

— Если значения атрибутов, по которым выполняется соединение отношений в формуле (3.1), не определены в некоторых кортежах БД, то эти кортежи будут отсутствовать в *TJ*, а вместе с ними значения мер и размерностей из этих кортежей. Если соединить меньшее количество отношений контекста, то значения мер и размерностей появятся в реализации *TJ*. Такие кортежи будем называть остатком соединения. Пользователь должен иметь возможность управления остатками.

4. Формирование гиперкубического представления

4.1. Реализация контекста. В качестве реализации контекста будем использовать представление данных в виде таблицы соединений, являющейся модификацией представления данных [18]. Совокупность свойств этой таблицы, которые будут рассмотрены ниже, позволяет получить необходимое представление для формирования многомерных представлений данных.

Рассмотрим преобразование представления реляционной БД: $\Re = \{R_1, R_2, ..., R_k\}$ в таблицу соединений *TJ* со схемой (S, g), где *S* — схема, определенная на множестве атрибутов $U = \{A_1, A_2, ..., A_n\}$; *g* — вектор вхождения кортежей отношений длины *k*.

Определим принцип формирования кортежей $t \in s$, где s — реализация (множество кортежей) таблицы TJ.

Рассмотрим все возможные сочетания без повторений отношений $R_1, R_2, ..., R_k$, удовлетворяющие свойству СБПИ. Пусть множество отношений $C' = \{R_{mas(1)}, R_{mas(2)}, ..., R_{mas(p)}\}$ — контекст, где mas — целочисленный массив из p номеров отношений текущего сочетания: c' — его реализация, ограниченная операцией селекции σ_F с логической формулой F:

$$c' = \sigma_F (R_{mas(1)} [W_{mas(1)}] \bowtie R_{mas(2)} [W_{mas(2)}] \bowtie \ldots \bowtie R_{mas(p)} [W_{mas(p)}]), \quad (4.1)$$

где $V_{mas(i)} \subseteq W_{mas(i)}$, $i = 1, 2, ..., p$, $V_{mas(i)}$ определены по формуле (3.1).

Для каждого кортежа $u \in c'$ формируем кортеж t по следующим правилам: $t[A_j] = u[A_j]$, если атрибут A_j принадлежит хотя бы одному отношению соединения, и $t[A_j] = emp$ в противном случае, где emp — пустое значение.

Каждому кортежу поставим в соответствие битовый вектор $g(t) = (g_1(t), g_2(t), ..., g_k(t))$, где $g_j(t) = 1$, если отношение R_j участвует в текущем соединении, и $g_j(t) = 0$ в противном случае.

Рассмотрим отношение частичного порядка над кортежами $t \in s$ [10].

Определение 4.1. Кортеж $t \in s$ является менее определенным или равным кортежу $t' \in s$, когда для любого атрибута A_i выполнено условие: если $t[A_i] \neq t'[A_i]$, то $t[A_i] = emp$ и $g_j(t') \geq g_j(t)$, $j = 1,\ldots,k$, причем $t[A_i] = t'[A_i]$, если A_i принимает значение $NULL$ в обоих кортежах. В этом случае будем писать $t \prec t'$, назовем кортеж t подчиненным кортежу t' и оба этих кортежа будем считать сравнимыми.

В представлении s достаточно хранить только кортеж t', который содержит в себе все менее определенные либо равные кортежи. Следовательно, следующим этапом построения представления s является удаление в нем всех подчиненных кортежей. Равенство неопределенных значений в определении 4.1 позволяет избавиться от подчиненных кортежей t, которые получены из тех же кортежей БД, что и t'. Отличие значений $NULL$ и emp в том, что первое указывает на неопределенное значение атрибута, а второе — на отсутствие соответствующего кортежа в текущем соединении. Очевидно, что отношение $\prec$ является транзитивным.

Заключительным этапом построения представления s является удаление в нем кортежей, для которых $F(t)=FALSE$. При этом кортежи t', для которых $t' \prec t$ и $F(t')=TRUE$, считаются лишними, поскольку неопределенные данные в t' доопределяются в t и пользователь от них отказался.

Определение 4.2. Проекция $\pi_{R(L)}(s)$ есть совокупность кортежей $u[R(L)]$, определенных на множестве всех атрибутов отношений $R(L)$, где для каждого $u[R(L)]$ существует кортеж $t \in s$ — такой, что $u[R(L)] = t[R(L)]$ и $g_{mas(i)}(t) = 1$, $i = 1,2,\ldots,p$.

Логическое ограничение $F(t)$ для каждой размерности будем представлять в виде дизъюнктивной нормальной формы:

$$F = F_1 \vee F_2 \vee \ldots \vee F_m.$$

Здесь каждая формула F_i является конъюнктом:

$$F_i = F_{i,1} \wedge F_{i,2} \wedge \ldots \wedge F_{i,q},$$

где $F_{i,j}$ — предикат сравнения языка SQL.

Если какой-либо предикат $F_{i,j}$ не определен на кортеже t, то он аннулируется — заменяется значением $TRUE$, если в этом конъюнкте еще есть не аннулированные предикаты, в противном случае — $FALSE$. Такая подстановка позволяет оставить в s кортежи, для которых пока не определены некоторые атрибуты или отсутствуют связанные по значениям кортежи в других отношениях, что также является предметом анализа информации. Формула F после подстановки будет принимать только два значения: $TRUE$ и $FALSE$.

4.2. Реализация гиперкубического представления. Под реализацией представления гиперкуба будем понимать множество таблиц соединения s_i и таблиц, сформированных по псевдоконтекстам. Далее эти таблицы будем называть исходными. Это соответствует поликубическому представлению данных (кубоидам). Представления, необходимые для анализа данных, могут быть получены:

— непосредственно из исходных таблиц,

— с использованием рассмотренных операций проекции по атрибутам [8, 9] и проекции по контексту в соответствии с определением 4.2,

— с использованием определения дополнительных операций.

Рассмотрим применение таблиц для формирования результирующего представления данных GC, используемого при проведении различных видов многомерного анализа. Совокупность таблиц данных — T_i, $i = 0, 1, 2,\ldots, d$, определенных на множествах атрибутов размерностей D_i соответственно. Таблица T_0 соответствует контексту приложения и определена на множестве атрибутов $D_0 = D_1 \cup D_2 \cup \ldots \cup D_d \cup M$. При этом

$$T_0 = s_0[D_0], \quad (4.2)$$

где s_0 — таблица соединения для контекста приложения C_0; таблица T_0 содержит остатки соединения и значения мер, сопоставленные значениям размерностей.

Для формирования каждой размерности ($1 \le i \le d$) в зависимости от потребностей пользователя используется одна из трех следующих формул:

$$T_i = s_i\,[D_i], \tag{4.3}$$

где s_i — таблица соединения для контекста C_i; таблица T_i содержит остатки соединения.

$$T_i = \pi_{R\cdot(L)}(s_0)[D_i], \tag{4.4}$$

где $C_i = \{R_1^*, R_2^*, \ldots, R_q^*\}$ — пустой контекст; L — вектор номеров отношений пустого контекста; таблица T_i не содержит остатков соединения.

$$T_i = \sigma_{F_i}(R'_1[W_1] \bowtie R'_2[W_2] \bowtie \ldots \bowtie R'_p[W_p])[D_i], \tag{4.5}$$

если $C_i = \{R'_1, R'_2, ..., R'_p\}$ — псевдоконтекст, $0 < i \le d$, F_i — логическое ограничение на кортежи, остальные обозначения и ограничения совпадают с формулой (3.1), таблица T_i не содержит остатков соединения.

Вывод. Предложенная модель многомерных данных является обобщением известных моделей, прежде всего за счет снятия ограничения (1.1). Технология ориентирована на работу аналитика, где не требуется быстрая (за доли секунд) реакция системы на запросы, поскольку в большинстве случаев аналитик должен вдумчиво выполнить различные виды анализа над различными представлениями.

Методологическая основа данного исследования может быть представлена следующим образом: операционная база данных должна удовлетворять принципам независимости, неизбыточности, непротиворечивости и т. д. Эта база данных является ядром приложений для множества пользователей, а не только отдельно взятого аналитика.

Библиографический список

1. Vassiliadis, P. A survey of logical models for OLAP databases / P. Vassiliadis, T. Sellis // SIGMOD Record. — 1999. — Vol. 28, № 4. — P. 64–69.

2. Pedersen, T.-B. A foundation for capturing and querying complex multidimensional data / T.-B. Pedersen, C.-S. Jensen, C.-E. Dyreson // Information Systems Frontiers. — 2001. — Vol. 26, № 5. — P. 383–423.

3. Lechtenborger, J. Multidimensional normal forms for data warehouse design / J. Lechtenborger, G. Vossen // Information Systems Frontiers. — 2003. — Vol. 28, № 5. — P. 415–434.

4. Progressive ranking of range aggregates / H.-G. Li [et al.] // Data & Knowledge Engineering. — 2007. — Vol. 63, № 1. — P. 4–25.

5. Lehner, W. Normal forms for multidimensional databases / W. Lehner, J. Albrecht, H. Wedekind // Proceedings of the Tenth International Conference on Scientific and Statistical Database Management. — Los Alamitos, 1998. — P. 63–72.

6. Giorgini, P. Goal-oriented requirement analysis for data warehouse design / P. Giorgini, S. Rizzi, M. Garzetti // In Proceedings of the 8th ACM international Workshop on Data Warehousing and OLAP: DOLAP '05. — Bremen, 2005. — P. 47–56.

7. Mazon, J. Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms / J. Mazon, J. Trujillo, J. Lechtenborger // Data & Knowledge Engineering. — 2007. — Vol. 63, № 3. — P. 725–751.

8. Ullman, J. Principles of database systems / J. Ullman. — Stanford : Computer Science Press, 1980. — 379 p.

9. Maier, D. The theory of relational databases / D. Maier. — Rockville : Computer Science Press, 1983. — 637 p.

10. Zykin, S. V. Formation of Hypercube Representation of Relational Database / S. V. Zykin // Programming and Computer Software. — 2006. — Vol. 32, № 6. — P. 348–354.

11. Зыкин, С. В. Динамические контексты базы данных реляционного типа / С. В. Зыкин // Информатика и ее применения. — 2014. — Т. 8, № 1. — С. 77-88.

12. Редреев, П. Г. Построение иерархий в многомерных моделях данных / П. Г. Редреев // Известия Саратовского университета. — 2009. — Т. 9, № 4, ч. 1. — С. 84–87. — (Математика. Механика. Информатика).

13. Casanova, M. Inclusion Dependencies and Their Interaction with Functional Dependencies / M. Casanova, R. Fagin, C. Papadimitriou // Journal of Computer and System Sciences. — 1984. — № 28 (1). — P. 29–59.

14. Missaoui, R. The Implication Problem for Inclusion Dependencies: A Graph Approach / R. Missaoui, R. Godin // SIGMOD Record. — 1990. — Vol. 19, № 1. — P. 36–40.

15. Levene, M. Justification for Inclusion Dependency Normal Form / M. Levene, M.-W. Vincent // IEEE Transactions on Knowledge and Data Engineering. — 2000. — Vol. 12, № 2. — P. 281–291.

16. Aho, A.-V. Data Structures and Algorithms / A.-V. Aho, J.-E. Hopcrof t, J.-D. Ullman. — Reading : Addison-Wesley, 1983. — 427 p.

17. Miller, L. Data Warehouse Modeler: A CASE Tool for Warehouse Design / L. Miller, S. Nila // Thirty-First Annual Hawaii International Conference on System Sciences. — 1998. — Vol. 6. — P 42–48.

18. Zykin, S. V. Automation of the interface formation between multidimensional and relational representation of the data / S. V. Zykin // Relational Databases and Open Source Software Development / Ed. J.-R. Taylor. — New York : Nova Science Publishers, 2010. — Chapter 2. — P. 43–66.

References

1. Vassiliadis, P., Sellis, T. A survey of logical models for OLAP databases. SIGMOD Record, 1999, vol. 28, no.4, pp. 64–69.

2. Pedersen, T.-B., Jensen, C.-S., Dyreson, C.-E. A foundation for capturing and querying complex multidimensional data. Information Systems Frontiers, 2001, vol. 26, no. 5, pp. 383–423.

3. Lechtenborger, J., Vossen, G. Multidimensional normal forms for data warehouse design. Information Systems Frontiers, 2003, vol. 28, no. 5, pp. 415–434.

4. Li, H.-G., et al. Progressive ranking of range aggregates. Data & Knowledge Engineering, 2007, vol. 63, no. 1, pp. 4–25.

5. Lehner, W., Albrecht, J., Wedekind H. Normal forms for multidimensional databases. Proceedings of the Tenth International Conference on Scientific and Statistical Database Management. Los Alamitos, 1998, pp. 63–72.

6. Giorgini, P., Rizzi, S., Garzetti, M. Goal-oriented requirement analysis for data warehouse design. In Proceedings of the 8th ACM international Workshop on Data Warehousing and OLAP: DOLAP '05. — Bremen, 2005. — P. 47–56.

7. Mazon, J., Trujillo, J., Lechtenborger, J. Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms. Data & Knowledge Engineering, 2007, vol. 63, no. 3, pp. 725–751.

8. Ullman, J. Principles of database systems. Stanford: Computer Science Press, 1980, 379 p.

9. Maier, D. The theory of relational databases. Rockville: Computer Science Press, 1983, 637 p.

10. Zykin, S.V. Formation of Hypercube Representation of Relational Database. Programming and Computer Software, 2006, vol. 32, no. 6, pp. 348–354.

11. Zykin, S.V. Dinamicheskie konteksty bazy dannykh relyatsionnogo tipa. [Dynamic contexts of relational-type database.] Informatics and Applications, 2014, vol. 8, no. 1, pp. 77-88 (in Russian).

12. Redreev, P.G. Postroenie ierarkhiy v mnogomernykh modelyakh dannykh. [Construction of hierarchies in multidimensional data models.] Izvestiya of Saratov University. Series Mathematics. Mechanics. Informatics. 2009, vol. 9, no. 4, part 1, pp. 84–87 (in Russian).

13. Casanova, M., Fagin, R., Papadimitriou, C. Inclusion Dependencies and Their Interaction with Functional Dependencies. Journal of Computer and System Sciences, 1984, no. 28 (1), pp. 29–59.

14. Missaoui, R., Godin, R. The Implication Problem for Inclusion Dependencies: A Graph Approach. SIGMOD Record, 1990, vol. 19, no. 1, pp. 36–40.

15. Levene, M., Vincent, M.-W. Justification for Inclusion Dependency Normal Form. IEEE Transactions on Knowledge and Data Engineering, 2000, vol. 12, no. 2, pp. 281–291.

16. Aho, A.-V., Hopcrof t, J.-E., Ullman, J.-D. Data Structures and Algorithms. Reading: Addison-Wesley, 1983, 427 p.

17. Miller, L., Nila, S. Data Warehouse Modeler: A CASE Tool for Warehouse Design. Thirty-First Annual Hawaii International Conference on System Sciences, 1998, vol. 6, pp. 42–48.

18. Zykin, S.V. Automation of the interface formation between multidimensional and relational representation of the data. Taylor, J.-R., ed. Relational Databases and Open Source Software Development. New York: Nova Science Publishers, 2010, Chapter 2, pp. 43–66.

Поступила в редакцию 03.11.2015
Сдана в редакцию 03.11.2015
Запланирована в номер 23.03.2016